Computer-Aided Design Package for Designers of Digital Optical Computers

Progress Report for Grant #N00014-90-J-4018 for Period 11/1/91 - 1/31/92

Principal Investigator: Miles Murdocca
Department of Computer Science
Rutgers University, Hill Center
New Brunswick, NJ 08903
(908) 932-2654
murdocca@cs.rutgers.edu

February, 1992

Summary

This report covers progress for the period 11/1/91 - 1/31/92 on a jointly sponsored AFOSR/ONR grant to Rutgers University that supports research in architecture and design of digital optical computers. Progress for this reporting period includes the ongoing development of an interactive design tool for digital optical circuits, and the development of optical interconnection methods that affect the design process. Prior to this report, an emphasis has been placed on the Bell Labs style of digital optical processor, in which arrays of optical logic gates are interconnected in free space with regular patterns at the gate level. The computer-aided design (CAD) tools and the optical interconnection methods that we have been developing under AFOSR/ONR support have evolved to the extent that we are now characterizing tradeoffs between the complexity of the optical interconnects and the complexity of the computer architecture.

The next section describes progress made on extending the existing design tools in order to study how the complexity of the optical interconnects influence the computer architecture. The section that follows summarizes progress made in exploring optical tradeoffs that affect the design process. As of this quarter, we are now in a position to characterize how changes in the optical interconnects affect computer architecture. One tradeoff that is of particular interest is in how the regularity of the interconnects affect the complexity of the optics, and the depth and breadth of the resulting circuits.

A few of the lessons learned thus far are:

Lesson #1: Regular interconnects, such as crossovers, can be maintained at every level of the computer architecture, from the logic gates all the way up to the system level.

Lesson #2: We have found that the expense of maintaining strict regularity can introduce significant cost in circuit depth and in circuit breadth. It appears, from initial studies by PhD students Gupta and Majidi using the newly modified tools, that a few irregular connections placed in selected positions in a circuit can have a significant impact on the size of the circuit being designed.

Lesson #3: A greater fan-out than two, such as four, can have a significant impact on reducing circuit depth. A superlinear improvement is possible for the regular interconnect model, since the increased fan-out eliminates problems associated with blocking.

Lesson #4: Irregular interconnects can be achieved with diffractive optical elements. However, studies by Stone (see below) show that there is a tradeoff between lens size and propagation distance. A completely irregular interconnect will effectively require a separate imaging system for each optical signal, and the resulting propagation distance of a few millimeters may not allow for steep angles of incidence, thus complicating the realization of a completely irregular interconnect. A mix of regular and irregular interconnects appears to be a reasonable compromise when the tradeoffs among the optics and architecture are considered together. One rule of thumb that we are exploring is to use regular interconnects for clusters of signals, 16x16 for example, and then to use irregular interconnects between clusters. In this way, propagation distance can be increased while simultaneously reducing circuit depth that is attributed to the regularity.

Interactive Design Tool for Digital Optical Circuits

Our previous work in designing digital optical circuits is based on an architecture that uses arrays of optical logic gates and regular free-space interconnects at the gate level such as perfect shuffles, banyans and cross-overs. By suitably masking connections, Boolean functions can be implemented. Figure 1 shows a circuit that implements a 3-to-8 decoder (lightly shaded lines indicate masked connections).

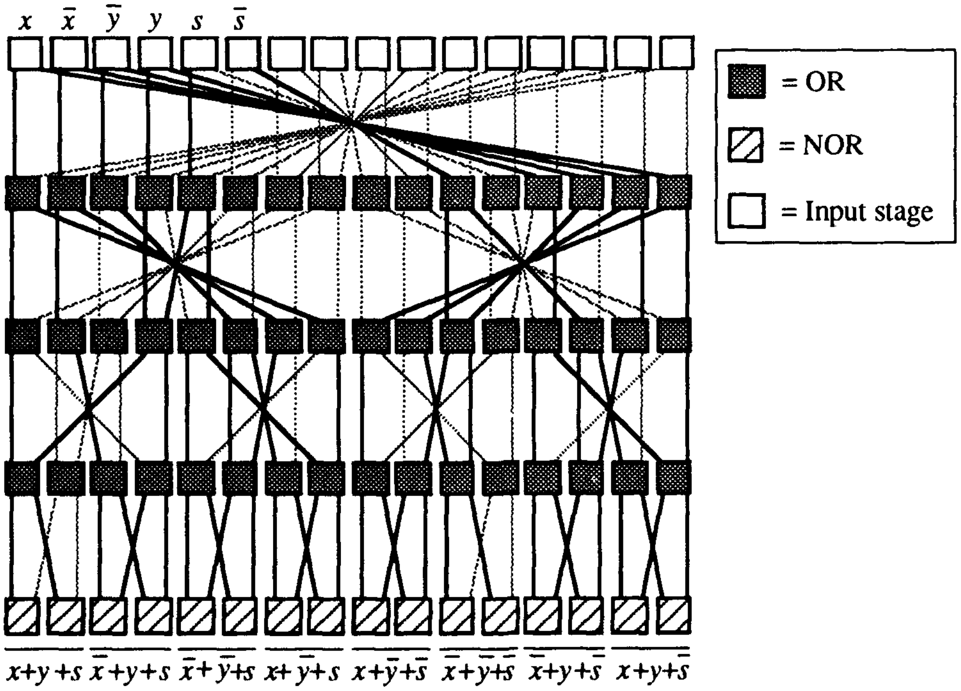

x $\bar{x}$ $\bar{y}$ y s $\bar{s}$
■ = OR
▨ = NOR
□ = Input stage
$\overline{x+y+s}$ $\overline{\bar{x}+y+s}$ $\overline{\bar{x}+\bar{y}+s}$ $\overline{x+\bar{y}+s}$ $\overline{x+\bar{y}+\bar{s}}$ $\overline{\bar{x}+\bar{y}+\bar{s}}$ $\overline{\bar{x}+y+\bar{s}}$ $\overline{x+y+\bar{s}}$

Figure 1: *A 3-to-8 decoder circuit is implemented by masking connections between optical gates. Flow of signals is from the top to the bottom.*

Statement A per telecon
Dr. William Miceli ONR/Code 1164
Arlington, VA 22217-5000

NWW 3/16/92

By maintaining strict regularity at the gate level, the only flexibility that a designer has is in choosing which connections to mask and which connections to leave unmasked. While algorithms exist to automatically generate masks for some basic circuits, good algorithms for generating general circuits do not exist. The circuit shown in Figure 1 is a 3x8 decoder which produces all possible combinations of the three input variables *x*, *y*, and *z*. A recursive algorithm for producing a general decoder is described in Ref. [1]. An unreported algorithm created by Gupta for generating multiplexors is also available. However, there are many circuit design problems, such as in connecting small optical circuits to form bigger circuits, for which the only known algorithms employ an exhaustive search of all possible combinations of masked and unmasked connections. For situations such as this, in which the essence of a good design cannot be captured by an algorithm, the approach we use is to allow an expert to create a design interactively. Although interaction is needed at some level of design, it does not make sense to involve the designer in those aspects of design for which good layout algorithms are known. Thus, the interactive design tool described below was created so that a designer can draw on as much automatic layout as is currently known, and then fill in the gaps interactively.

Xopid is an interactive design tool created by Gupta, which is based on an earlier tool developed by Majidi. The tool allows logic gates to have fan-ins and fan-outs that vary, and allows circuits to have irregular interconnections between gates. These features allow us to study the trade-offs involved when fan-in/fan-out values higher than two are used and when connections are not constrained to be regular. Other issues being studied with *xopid* include functional decomposition and PLA tiling.

In more detail, *xopid* is a menu-driven tool that allows the user to draw and manipulate digital optical circuits interactively in an X window. The user interface to *xopid* is shown in Figure 2. Five vertically stacked windows comprise the display area: the command window, the file-label window, the main drawing window which contains horizontal and vertical scrollbars, the help window, and the message window. The command window contains buttons which the user selects for different circuit manipulation commands. When a button is clicked, the button is highlighted, and a brief message describing its function is displayed in the help window. If the execution of a user command results in an error or some other exceptional behavior, an appropriate message is displayed in the message window. The file-label window displays the name of the circuit being currently manipulated.

A synopsis of the functions available to the user are described below, which is taken from the manual pages for the software:

NEW
Clears the current circuit.

LOAD...
Prompts the user to specify a *.cir* file (a circuit file, stored in *xopid* format). The circuit described in this file then becomes the new current circuit. If the specified file does not exist, the empty circuit becomes the new current circuit.

MERGE...
Prompts the user to specify a *.cir* file. The circuit in this file is merged into the current circuit at a position that the user can specify by clicking. The merge operation fails if a circuit overlap would be created.

SAVE
Saves the current circuit in the file named by extending the filename displayed in the file-label window with a *.cir* extension.

SAVE AS...
Prompts the user to specify a *.cir* file. The current circuit is then saved in this file.

PRINT
Prints the current circuit in PostScript format in the file named by extending the filename displayed in the file-label window with a *.ps* extension.

REFRESH
Redraws the circuit on the pixmap that is displayed in the main drawing window.

FLIP
Waits for the user to specify a rectangular region by pressing the left mouse button on the upper left corner, dragging the pointer to the lower right corner and releasing the left mouse button. A copy is made of the sub-circuit corresponding to the user-specified rectangular region, which is flipped along a vertical axis passing through the center of the region and stored in *.Clipboard.cir* from where it can be pasted using the paste option.

COPY
Similar to *FLIP* except that the sub-circuit is not flipped before it is stored in *.Clipboard.cir*.

CUT
Similar to *COPY* but deletes the sub-circuit corresponding to the user-specified region from the current circuit.

PASTE
Waits for the user to specify a point, where the upper left corner of the circuit stored in *.Clipboard.cir* is merged into the current circuit, provided this does not result in an overlap.

QUIT
Exit gracefully from *xopid*, discarding the current circuit.

OR/NOR/AND/NAND
Waits for the user to specify a rectangular region as described in *FLIP* and fills the rectangular region with gates of the type displayed in the help-window. If a gate already exists in the region, its type is changed to that displayed in the help-window. The user can toggle through these gate types in a circular fashion by repeatedly clicking this command button.

BUTTERFLY/SHUFFLE/CROSSOVER
Waits for the user to specify a rectangular region as described in *FLIP* and inserts masked connections corresponding to the chosen pattern between gates in this region. The user can toggle through these regular interconnection patterns in a circular fashion by repeatedly clicking this command button.

CONNECT/DISCONNECT
Waits for the user to press the left mouse button over one gate, drag the pointer till it is over another gate, and release the button. If the *CONNECT* option is active, a new masked connection is made between an output of the first gate and an input of the second gate if one does not exist already. If the *DISCONNECT* option is active, the existing connection, if any, between an output of the first gate and an input of the second is removed. The active option is displayed in the help-window. The user toggles between these two options by clicking on this command button.

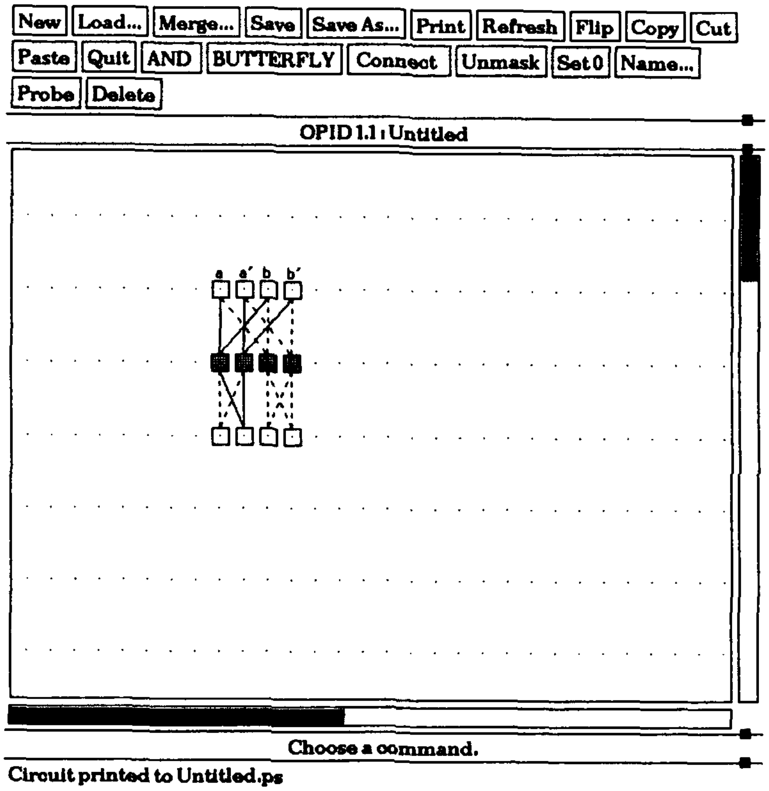
UNMASK/MASK
Waits for the user to specify two gates as described in *CONNECT/DISCONNECT*. If the *UNMASK* option is active, a path of connections, if one exists, leading from the output of the first gate to the input of the second gate is found and all connections on this path are unmasked. If the *MASK* option is active, all connections on the path are masked. The active option is displayed in the help-window. The user toggles between these two options by clicking on this command button.

SET 0/SET 1/UNSET
Waits for the user to click on a gate. If the *SET 0* option is active, the output of the gate that is selected is set to zero. If the *SET 1* option is active, the output of the gate that is selected is set to one. If the *UNSET* option is active, any Boolean value to which the output of the gate that is selected had been tied is removed. The active option is displayed in the help-window. The user toggles between these options by clicking on this command button.

NAME...
Prompts the user to specify a name for a gate and waits for the user to click on a gate. The output signal of the gate that is selected is then given the specified name. If a name is not specified, and if the output of the gate already has a name, that name is removed.

PROBE
Waits for the user to select a gate. The output value being generated at the gate and the Boolean expression representing its output are displayed in the message window.

DELETE
Waits for the user to specify a rectangular region as described in *FLIP*. All gates that lie inside this region are deleted from the current circuit as well as all connections that are incident on any gate inside this region.



OPID 1.1 : Untitled

Choose a command.

Circuit printed to Untitled.ps

Figure 2: *The user interface to* xopid.

Optical Interconnects

Progress continues on Stone's studies into how conventional and diffractive optics can be used to solve interconnect problems in computing. Stone has continued studying the characteristics of birefringent materials for spot-array generation and optical interconnection. The achromatic nature of birefringent materials is of particular interest when compared with diffractive array generators. A paper was presented on this topic during the November 3-8 Annual Meeting of the Optical Society of America in San Jose.

A study has continued into tradeoffs between approaches using micro-optics and macro-optics for interconnecting arrays of optical logic devices. Results of these analyses are reported in a book chapter that has been prepared for *Optical Computing Hardware*, edited by Sing Lee and Jürgen Jahns (see References below). An excerpt from this study is given below:

> *Critical Distance for Collimated Array*
>
> An example of the diffraction-based trade-offs in device spacing and propagation distance is given for the case involving a collimated array of beams. The critical distance is a function of the microlens diameter. For example, consider an array with a device spacing $\Delta = 200\mu m$ and light of wavelength $0.85\mu m$. If the lens is only $10\mu m$ in diameter ($D/\Delta =$

0.05), there will be a buffer zone of width B = 95μm on each side of the microlens over which the light may spread before crossing into the neighboring channel. The diffraction spread angle of the beam from such a small lens, however, is large (4.9°), and after only L_c = 1.1mm the beam begins to spread beyond the 95μm buffer and mix with the neighboring signal. Similarly, as the microlens diameter approaches the gate spacing, the critical distance L_c is also very small. Near this other extreme, if D/Δ = 0.95 (D=190μm) the diffraction angle is a much smaller .26°, but the buffer zone width is now reduced to B=5μm, and L_c is again only 1.1 mm. However, for less extreme values of D/Δ (*e.g.*, near 0.5), L_c is much larger (nearly 6mm). Figure 3 shows a plot of L_c (given in millimeters) as a function of varied fill factor D/Δ and gate spacing Δ. Since the diffraction angle decreases with increasing lens diameter, one might suspect that low crosstalk could be maintained over longer distances if the full width Δ could be utilized for the microlens apertures. Effective use of these larger apertures can be accomplished by slightly focusing the beam emerging from the microlens, thus avoiding the condition in which any spreading of the collimated beam from a lens with D = Δ results in crosstalk. This focused array configuration is discussed in the book chapter, as well as several other results of Stone's study.

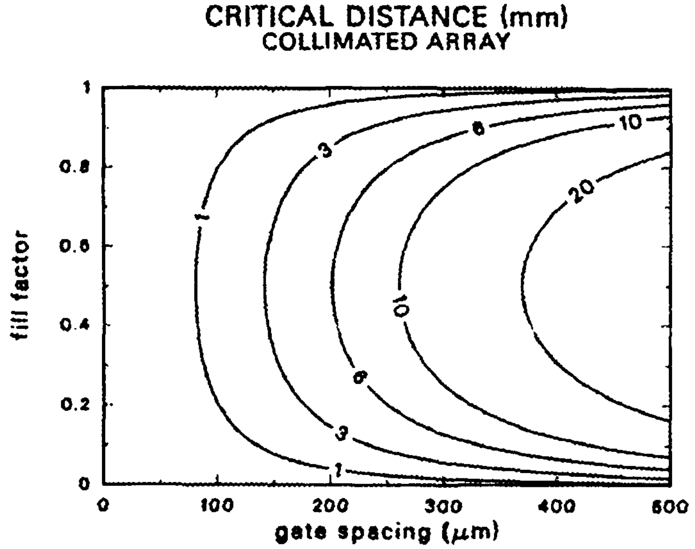


Figure 3: *Plot showing critical distance as a function of gate spacing and fill factor.*

The micro/macro-optics study has been useful in identifying niches that are best served with conventional optics, and those where diffractive optics are more reasonable to apply. This aspect of Stone's work influences the design of digital circuits, and we are using the *xopid* tool to investigate the architectural implications of using various combinations of micro and macro-optics.

The perspective of classic lens design is an important tool with which to study both conventional and diffractive optics, and to that end, the OSLO Series 2 lens design program was purchased using cost sharing funds which were provided by Rutgers for this grant. The software is important as a research tool for studying the properties of new elements and new configurations.

For example, it has the flexibility for the user to introduce routines describing the characteristics of novel or unusual optical surfaces. This software was installed during this reporting period, and Stone is exploring the features of the software.

Our research project is being carried out in the Computer Science Department at Rutgers University, where not all of the computer scientists have a sufficient background in optics in order to understand the tradeoffs. A regular series of seminar meetings was organized by Stone in order to tutor the members of this research group, in which guiding principles of physical and geometrical optics were discussed along with architectural issues in computing.

Miles Murdocca

References

[1] Murdocca, M. J., A. Huang, J. Jahns, and N. Streibl, "Optical Design of Programmable Logic Arrays," *Applied Optics*, **27**, pp. 1651-1660, (May 1, 1988).

[2] D. Smith, M. Murdocca, and T. Stone, "Parallel Optical Interconnections," book chapter in *Optical Computing Hardware*, vol. 2, edited by S. Lee and J. Jahns, Academic Press, (1992), to appear.